SNNs Are Not Transformers (Yet): The Architectural Problems for SNNs in Modeling Long-Range Dependencies

William Fishell[1] and Suraj Honnuraiah[2]

[1]Columbia University, wf2322@columbia.edu
[2]Harvard University, suraj_honnuraiah@hms.harvard.edu

Abstract

Spiking Neural Networks (SNNs) have gained increasing attention for their ability to operate on low-power neuromorphic hardware, offering a pathway toward energy-efficient large language models (LLMs). However, they continue to underperform state-of-the-art artificial neural networks (ANNs) on tasks such as language modeling, limiting their broader adoption. Here, we present an explicit covering-number analysis of SNNs, focusing on the non-leaky integrate-and fire (nLIF) model. Building on recent advances in causal-piece analysis and local Lipschitz continuity, we derive a global Lipschitz constant for nLIF networks and extend this framework from single-token inputs to full sequences. Our analysis reveals that the sample complexity of nLIF architectures grows quadratically with sequence length, in contrast to the linear or logarithmic scaling observed in recurrent and Transformer models. These findings expose a fundamental architectural limitation in SNNs and suggest targeted architectural and biological modifications needed to overcome it. Finally, we show that the same assumptions emerge in biological circuits, emphasizing the critical role of inhibitory interactions in enabling efficient learning of long-range dependencies.

Introduction

References to biology have been important in pushing the bounds of machine learning models. From the application of the Hebbian learning rule in Hopfield networks to the convolutional filters that emulate local receptive fields in the visual cortex, modern deep-learning models continually draw practical inspiration from biological systems. Although strict biological realism isn't required—many innovations like self-attention and gradient descent have no direct counterpart in real neurons—grounding our algorithms in biological principles offers clear benefits: by drawing on decades of neuroscience research, we can uncover efficient mechanisms for learning and adaptation, and by emulating evolution's optimizations for energy use, robustness, and adaptability, we can design models that are both powerful and resource-efficient.

Spiking Neural Networks (SNN)—often called the "third generation" of neural models—offer greater biological realism than standard artificial neural networks (ANN). These networks achieve remarkable energy efficiency by emitting sparse, temporally distributed spikes—much like biological action potentials. Their built-in sparsity and sequential processing stand in stark contrast to the $O(N^2)$ complexity of Transformer self-attention and the billions of operations required during large-language-model inference. SNNs have already achieved promising results, such as SpikeGPT, which achieves performance comparable to GPT-3 while using 33 times less power [1]. Furthermore, these networks leverage both temporal and spatial dependencies and include a richer set of programmable parameters—such as delays and neuron-specific thresholds—which endows them with greater expressiveness than recurrent neural networks of the same depth and width [2, 3].

Despite the potential for increased expressivity, SNNs struggle from many of the same limitations that RNNs have: a finite memory horizon [4], and the inability to model non-sequential interaction [5]. These issues in SNNs have caused them to struggle with modeling long-term dependencies. Creating wider adoption of these networks requires devising modifications that solve these issues. Inspired by Edelman et al. [6], we use a learning-theoretic approach to understand how learning varies with sequence length. Measuring the change in the number of samples required to achieve PAC learnability as a function of the sequence length of those samples indicates how well the SNN architecture models long-range dependencies. While prior work on the VC-dimension of SNNs has established foundational capacity bounds [7, 2, 8, 9], it has largely overlooked the role of sequence length and has not explored covering-number approaches. In this work, we leverage recent results on the local Lipschitz continuity of SNNs [10] to derive a covering-number bound, establishing a novel relationship between sample complexity and sequence length. Our analysis provides new theoretical insights into SNNs and helps explain their current limitations in language modeling tasks compared to Transformers and RNNs.

An SNN processes information as time-stamped spike trains instead of static real-valued activations. Each neuron integrates incoming weighted spikes from the preceding layer, accumulating a membrane potential: this potential decays continuously. When it exceeds a fixed threshold, the neuron emits a spike and transmits its associated edge weights to downstream connections after a time delay Δt. This characteristic of the SNN results in sparse network activity. As each spike train passes through, only some of the neurons fire. Because the computation unfolds in continuous time, the network's state —and thus its output —depend on the moment of observation by the output layer. We focus on the non-leaky and leaky integrate-and-fire (nLIF, LIF) neuron models—the most widely adopted spiking neuron models. Although all SNNs share the pipeline sketched above, they diverge in their spike-encoding schemes and in the synaptic delays they assume. The LIF model captures these ingredients in their simplest canonical form and underlies many of the more sophisticated spiking architectures encountered in practice (for a more detailed overview of the specifics of SNNs and LIFs, see App. A). The nLIF is a simpler form of the LIF that does not have a leaky membrane potential (for a formal definition of nLIFs see App. C).

Results

1 Pseudo-Attention in the Leaky Integrate-&-Fire Model

The LIF architecture enables earlier tokens to influence later ones by effectively 'attending' to them and thus modifying the state of the network[1]. Although this mechanism is not identical to traditional attention, several similarities are noteworthy. At its simplest, an attention mechanism quantifies the similarity between pairs of tokens in the context. Suppose tokens c_1 and c_2 occur at positions i and $i+n$, so

$$t_i = i\,\Delta t, \quad t_j = (i+n)\,\Delta t.$$

We represent their embeddings as instantaneous spike-train vectors.

$$\mathbf{S}(t_i) = \big[S_1[i],\,\ldots,\,S_D[i]\big]^T, \quad \mathbf{S}(t_j) = \big[S_1[j],\,\ldots,\,S_D[j]\big]^T.$$

The scaled dot product does not occur in the LIF, but at $\mathbf{S}(t_i)$, the spike train raises the membrane potentials of its active subset of neurons, so when $\mathbf{S}(t_j)$ arrives, the neurons that represent these two tokens are more likely to fire. In contrast, if the overlap is small, fewer neurons will spike at time t_j representing their similarity. The state of the network at any time t is defined as a function of both the temporal dynamics and the contextual input, where the interactions between all pairs c_i and c_j are inherently present. Based on the described behavior, the LIF architecture strikes a balance between an RNN and a Transformer in expressing non-sequential interactions. Unlike an RNN, the LIF model supports some non-sequential computation. However, its inherent sequential nature — stemming from how tokens feed into the network as spike trains — makes it impossible to realize the attention mechanism fully. Furthermore, the introduction of distance between two tokens can introduce noise, which can affect the network state and obscure comparisons between tokens in the LIF architecture, thereby acting as a further barrier to attention in SNNs.

Conceptually, the LIF architecture can be viewed as having a kind of short-term memory. Because each neuron's membrane potential carries over across time steps, it retains a faint trace of recent activity. That carry-over creates limited non-sequential interactions, enough for nearby events to influence one another, though without forming true recurrence. In this sense, LIF neurons offer a weak analog of attention, providing selective temporal weighting without explicit dot-product computations, in contrast to the query–key matching used in Transformers. This produces short-range contextual effects that resemble attention kernels in their temporal weighting but remain strictly sequential in operation. Variability in spike timing with temporal distance can introduce noise that may obscure these effects, and a quantitative comparison with attention mechanisms is left for future work.

2 Sample Complexity of SNNs, Transformers, and RNNs

The VC dimension of a feedforward spiking neural network with K synaptic edges grows tightly on the order of $\Theta(K^2)$ [11, 8]. Intuitively, this quadratic scaling reflects the vast combinatorial flexibility afforded by programmable parameters—each weight, delay, and threshold setting contributes to carving out distinct decision regions, allowing the network to shatter on the order of K^2 input points. Although Maass' original proof focused on the nLIF model, the same argument holds for the standard LIF architecture: introducing additional programmable leak parameters can only maintain or increase the network's expressive capacity. Furthermore, this does not increase the VC dimension of the SNN, as the leaks function simply as another programmable parameter (for more details on why this is the case, see App. B). There is a strong understanding of the expressive capacity of the SNN, yet comparatively little research on how this capacity changes as the size of the inputs being modeled changes. Transformers and RNNs have well-understood bounds on how the number of samples required to achieve PAC learnability scales with the input sample size. RNNs require

[1]The same pseudo-attention phenomena in an LIF exist in an nLIF. We focus on the LIF to avoid redundancy.

linear growth in the size of an input in the number of samples to achieve PAC learnability as the sequence size grows. Transformers achieve an even more impressive $log(T)$ sample increase requirement for modeling longer sequences [6, 12][1]. Unlike Transformers, which have an inductive bias towards sparsity, SNNs have an inductive bias towards low-frequency functions [13]. This inductive bias indicates that they may perform poorly on modeling the relationship between a small set of important input tokens: the long-range dependency modeling task. To date, no analysis has quantified how SNN sample complexity grows with sequence length. In this work, we fill that gap by deriving explicit bounds for nLIF networks, revealing exactly how samples must scale as sequences get longer. These insights deepen our theoretical understanding of spiking architectures and help explain their inherent bias toward low-frequency mappings.

2.1 Non Leaky Integrate-&-Fire Covering Number

Under appropriate assumptions on our function class, we show that an nLIF SNN is globally Lipschitz continuous. We then derive the covering-number for both single-token and multi-token inputs and compare them to quantify how sample complexity scales with sequence length. Henceforth, we refer to the single-input-token setting as the Single Input Spike (SIS) problem and the multi-input-token setting as the Multi Input Spike (MIS) problem.

2.2 SIS Covering Number

The output spike time of an nLIF neuron can be made locally Lipschitz continuous by restricting attention to the specific subset of its inputs and weights that actually cause it to fire. From this, we show, with proper assumptions, the global Lipschitz constant and derive the covering number for the SIS case, and then extend to the MIS problem. Our work builds on work that proves local Lipschitz continuity with respect to weight and input spike time for an output spike time in an nLIF SNN. In deriving the covering number, we introduce common terminology used for the remainder of the paper.

A **causal piece**, denoted as P, for neuron i in layer ℓ is the set of all pre-synaptic spike-time vectors and weights that lead to its firing:

$$\left(t_i^{(\ell-1)}, W_i^{(\ell)}\right) \in R^{N_{\ell-1}} \times \mathcal{W}_i^{(\ell)} \quad \text{s.t. these inputs cause neuron } i \text{ in layer } \ell \text{ to spike.}$$

Its **causal set** is the set of pre-synaptic neuron spike times whose spikes precede its firing time:

$$C_i^{(\ell)}\left(t_1^{(\ell-1)}, \ldots, t_{N_{\ell-1}}^{(\ell-1)}\right) = \{\, j \mid t_j^{(\ell-1)} \leq t_i^{(\ell)} \}.$$

Lastly a **causal path**, denoted P_i^ℓ, for neuron i in layer ℓ is the set

$$P_i^\ell = \left\{ \left(t_j^1, \ldots t_j^{(\ell-1)}, W_j^1, \ldots W_j^{(\ell)}\right) \in R^{N_{\ell-1}} \times \mathcal{W}_i^{(\ell)} \;\middle|\; \text{these inputs cause neuron } i \text{ in layer } \ell \text{ to spike} \right\}.$$

Recent work shows that, when restricted to a fixed causal piece, the output spike-time map is locally Lipschitz continuous in those pre-synaptic times and weights[2] [10] (For a proof sketch and further details on this paper, see Appendix C).

Given a causal set for some neuron in layer ℓ that spikes at time T, the causal set is locally Lipschitz continuous with respect to its causal piece.

Let

$$a = \left(t_j^{(\ell-1)}, W_{ij}^{(\ell)}\right)_{j \in P | C_i^{(\text{out})}|}, \quad b = \left(t_j'^{(\ell-1)}, W'^{(\ell)}_{ij}\right)_{j \in P | C_i \ell^{(\text{out})}|}$$

[1] When we say RNN, we are referring to a vanilla RNN

[2] pre-synaptic here simply means the immediate previous layer. This work uses a telescoping method to extend the Lipschitz bounds

be two configurations of pre-synaptic spike times and weights on the causal piece ($P|C_L^{(\text{out})}|$ and satisfy

$$\big\|a-b\big\|_{L_\infty\left(P[C_i^{(\text{out})}]\right)} \;\le\; \delta \;<\; \min_{j\in P[C_i^{(\text{out})}]}\Big(\sum_{k\in P[C_i^{(\text{out})}]} W_{jk}^{(\ell)} - \vartheta\Big)$$

So that any perturbation of size at most δ neither delays an originally–causal pre-synaptic spike past the output time T nor advances a non-causal spike before T. The total weighted input stays above the threshold, so the neuron still fires at T. Then the output spike-time map $t_L^{(\text{out})} : (t^{(L-1)}, W^{(L)}) \mapsto t_L$ is locally Lipschitz continuous on that piece, with

$$\big\|\, t_i^{(\text{out})}(a) \;-\; t_i^{(\text{out})}(b)\big\|_{L_\infty\left(P[C_i^{(\text{out})}]\right)} \;\le\; 2\,|C_i^{(\text{out})}|\,\max\Big(\tfrac{\bar{W}}{\delta},\,\tfrac{\tau_s}{\delta}\Big)\,\big\|\,a-b\big\|_{L_\infty\left(P[C_i^{(\text{out})}]\right)} \tag{1}$$

where

$$\bar{W} = \max(W_{ij}^L)\ ,\ \tau_s = \text{Synaptic time Constant}$$

Assumption 2.1 (Assumptions)**.** *Let*

1. *the network be a feedforward nLIF SNN;*
2. *M be a set of training examples s.t.*

$$\forall\, n \in \text{nLIF},\ \exists\, m_i \in M :\ F(m_i) \implies \textit{neuron } n \textit{ spikes}, \quad \textit{where } F \textit{ is the network map;}$$

3. *each neuron's causal set C_n^ℓ is non-empty;*
4. *all synaptic weights w_{ij}^ℓ are positive and, for every causal piece P,*

$$\sum_{(i,j)\in P} w_{ij}^L \;-\; \vartheta \;>\; \delta > 0.$$

5. *the reset for a neuron's membrane after spiking is instantaneous*

Theorem 2.1 (Global Lipschitz Bound)**.** *Under Assumption 2.1, there exists a constant L_{global} s.t. for any two input-spike patterns a, b with $|a-b|_{L_\infty} \le \gamma$,*

$$\big|T_{\text{output}}(a) - T_{\text{output}}(b)\big|_{L_\infty} \;\le\; L_{\text{global}}\,|a-b|_{L_\infty}$$

Proof. Consider the output layer of neurons in the nLIF. By the assumption, each of the neurons in the output layer has some causal set C_i^ℓ that is not empty. Because $\exists$ a δ margin across all the thresholds and the causal set of each of these output neurons is non-empty, we can construct a Layer Lipschitz bound across this layer by taking the

$$L_{\text{outlayer}} \;=\; 2 * \max_i\Big(\Big|C_i^\ell\Big|\ \max\Big(\frac{\bar{W}}{\delta},\,\frac{\tau_s}{\delta}\Big)\Big).$$

thus any pre-synaptic inputs a,b $\in P(C_i^{output})$ s.t $|a-b|_{L_\infty} \le \delta$ are locally Lipschitz continuous for neuron n_i^ℓ s.t.

$$|T_i^\ell(a) - T_i^\ell(b)|_{L_\infty} \le L_{outlayer}|a-b|_{L_\infty}$$

This can be traced recursively through the layers towards the input by telescoping. Consider a, b are now input spikes in the causal piece of neuron $n_j^{\ell-1}$ and $a, b \in P_i^L$ s.t.

$$\big|T_\ell\big(T^{(\ell-1)}(a)\big) - T_\ell\big(T^{(\ell-1)}(b)\big)\big|_{L_\infty} \;\le\; L_\ell\,\big|T^{(\ell-1)}(a) - T^{\ell-1)}(b)\big| \;\le\; L_\ell * L_{\ell-1}\,|a-b|_{L_\infty}$$

with the new bounds on the input that $|a-b|_{L_\infty} \leq \frac{\delta}{L_\ell * L_{\ell-1}}$ where L_ℓ and $L_{\ell-1}$ are defined as the max Lipschitz constants of their respective layers. By continuing this process to the input layer, we can construct a global Lipschitz constant for the entire nLIF feedforward network.

$$L_{\text{Global}} = \prod_{\ell=1}^{D} \max_i \Big(\Big| C_i^\ell \Big| \max\Big(\frac{\bar{W}}{\delta}, \frac{\tau_s}{\delta} \Big) \Big).$$

Because any input spikes a, b are in some causal path and all neurons in the network are covered in some causal path s.t. L_{global} can bound the output spike times of that neuron, L_{global} is a global Lipschitz constant with respect to spike time and weight for the network. By simply setting our free term $\gamma = \frac{\delta}{L_{Global}}$, the differences in the inputs will not cause a change in any causal path, so the entire network is defined by the global Lipschitz constant L_{global}

$$|T_i^\ell(a) - T_i^\ell(b)|_{L_\infty} \leq L_{Global} |a-b|_{L_\infty}$$

□

Under our assumptions, the nLIF network is globally Lipschitz continuous in the spike-time domain. To extend this to real-valued inputs and outputs, we compose three Lipschitz maps: an encoder that converts each real input into a spike train, the nLIF network itself (which is Lipschitz in spike times), and a decoder that reads the output spike train and maps it back to the target space. Because the composition of Lipschitz maps remains Lipschitz, the end-to-end map from original inputs to final outputs is Lipschitz. This lets us apply standard covering-number bounds directly in the real-valued signal domain. To illustrate the simplicity of constructing such a composition, we use a toy example of classifying words (for the details of this Example see App. D).

Example D outlines a process of applying a time to first spike (TTFS) encoding and softmax decoding to the input and output spaces to create a global Lipschitz constant for the function. Using the composition of Lipschitz continuous functions and the results from **Theorem ??**, a function class F can be created from nLIF SNN, and a covering number can be constructed. Provided the assumptions in 2.1 are met. For each of these functions in the function class, there is a global Lipschitz constant defined as

$$\begin{aligned}
L^i_{\text{total}} &= L^i_m \, L^i_{\text{global}} \, L^i_o, \text{ where} \\
L^i_m &= \text{Lipschitz constant of the input encoder (e.g. TTFS + softmax)}, \\
L^i_{\text{global}} &= \text{Lipschitz constant of the nLIF network in the spike-time domain}, \\
L^i_o &= \text{Lipschitz constant of the output decoder (e.g. softmax)}.
\end{aligned}$$

By taking the $max(L^i_{global})$, a Lipschitz constant which covers the entire function class can be derived, denoted $L_{MaxTotal}$. Due to our restriction on distinguishing perturbations of at most γ, the covering number is

$$\log N(\gamma, F) \leq D \log(\frac{\Delta L_{MaxTotal}}{\gamma})$$

This bound enables us to control the error within the function, and thus, the sample complexity M becomes

$$M = O(D \log(L_{MaxTotal})$$

Crucially, L_{MaxTotal} scales with each layer's maximum causal-set size. Allowing neurons to fire multiple times (MIS) rather than just once (SIS) thus induces a pronounced jump in the resulting sample complexity, because the causal sets grow with respect to the number of input spikes.

2.3 MIS Covering Number

The MIS problem consists of a sequence of tokens inputted as a spike train, where each token follows the preceding element after some time delay. Building on **Example D**, we encode a sequence of tokens into spike times as follows:

$$\text{(Sequence-token temporal encoder)} \quad t_s\big(m_i^{(s),j}\big) = \frac{m_i^{(s),j} - \min(M^j)}{\max(M^j) - \min(M^j)}\, T \;+\; (s-1)\,\Delta t, \quad s = 1, \ldots, S,$$

This encoding scheme can be implemented using a Dirac delta function to fire pulses at the moments of interest, as encoded by the sequence-token temporal encoder.

We follow the SIS setup described in **Example D** by keeping T small enough that a single token can trigger at most one spike per neuron as it moves through the network. Over a sequence of tokens, however, a neuron may fire multiple times.

Theorem 2.2 (Global Lipschitz Bound for MIS)**.** *Under Assumption 2.1, there exists a constant* L_{globalS} *s.t. for any two multi-spike input patterns* a, b *of length* S*,*

$$\big|T_{\text{output}}(a) - T_{\text{output}}(b)\big|_{L_\infty} \;\le\; L_{\text{globalS}}\, |a-b|_{L_\infty}$$

Here, L_{globalS} *is the global Lipschitz constant for the sequence-based encoding of length* $|S|$*, and it satisfies*

$$L_{\text{global}S} = O\big(L_{\text{global}}^{|S|}\big)$$

Where L_{global} *is the global Lipschitz constant established for the SIS problem on the same nLIF SNN.*

This result can be extended by using a composition of Lipschitz-continuous functions to encode and decode the spike train, as in **Example D**. In particular, for any sequence $S \in M^S$, applying the nLIF under suitable input bounds yields a globally Lipschitz map

$$f(S) \;=\; \phi_{\text{out}}\big(nLIF(\phi_{\text{in}}(S))\big),$$

with constant L_{Func} s.t.

$$\big|f(S) - f(S')\big|_{L_\infty} \;\le\; L_{\text{Func}}\, \|S - S'\|_{L_\infty}, \text{ where } L_{func} = L_{out} * L_{in} * L_{GlobalS}$$

Theorem 2.3. *For any nLIF network for which we can establish global Lipschitz continuity, the worst-case sample complexity grows quadratically with the sequence length.*

Proof. Given a function class F) For the MIS problem, a global Lipschitz constant can be constructed for each of the f_i in the function class. By taking the maximum of these Lipschitz constants and invoking **Theorem 2.2**, the maximum global Lipschitz constant is equal to some $O(L_{Global}^{|S|})$ where L_{Global} is the constant for the SIS case. Thus, the covering number is

$$\log N(\gamma, F) \le D \times |S| \log\Big(\frac{\Delta O(L_{Global}^{|S|})}{\gamma}\Big)$$

When $|S|$ is the sequence length, applying standard log rules, $|S|$ can be brought down. Thus, the sample complexity for this problem is

$$M = O(D * |S|^2 \log(L_{Global}))$$

Controlling for the input dimension D (the size of each tensor) and the network's structure by increasing the sequence length, we need $|S|^2$ more samples to ensure PAC learnability. □

In an nLIF SNN, the D input channels arrive as an S-length spike train whose contributions aggregate into each neuron's membrane. In the MIS setting, the global Lipschitz constant compounds by a factor of L_{SIS} at each of the S spikes—growing as

$$\left(L_{\text{SIS}}\right)^{S}.$$

Since the covering-number bound also scales linearly with the DS-dimensional input space, this multiplicative Lipschitz blow-up combined with the dimension term yields the sample-complexity penalty.

$$O\big(\,S^2\big)$$

The quadratic sample-complexity growth both illustrates why nLIF SNNs struggle with long-term dependencies and shows that preventing explosive causal set expansion can improve their ability to model such dependencies. By preventing unbounded excitation, we can cap causal set expansion, thereby bounding quadratic sample-complexity growth.

Causal pieces are an active area of research, and it is currently not known whether causal pieces in SIS or MIS LIFs are locally Lipschitz continuous with respect to their causal piece. We leave the question of sample complexity for an LIF to a later paper, as further research is needed to understand how the causal set of a neuron grows when it is leaking.

3 Implications for Biological Neural Circuitry

The spiking dynamics of nLIF SNNs lead to weak theoretical guarantees for learning long-range dependencies. By mapping the architectural assumptions of these networks to their biological counterparts, we both demonstrate the biological plausibility of our framework and identify which additional neural mechanisms may be required in real circuits to achieve stronger learning guarantees.

3.1 Effect of Refractory Gating and Population Normalization

The refractory period following Na^+ activation prevents uncontrolled excitation in neural circuits ([14], [15]). Despite endogenous limits on neuronal firing, the sequential integration of inputs leads to quadratic scaling.

Lemma 3.1 (Refractory Period Constraint)**.** *Let $\tau_{ref} > 0$ be the refractory period enforcing $\Delta t_{\min} \geq \tau_{ref}$ between consecutive spikes of neuron i. For a sequence of length $|S|$ with inter-token delay Δt, the maximum spike count is:*

$$k_i^{\max} \leq \left\lfloor \frac{T}{\tau_{ref}} \right\rfloor = \left\lfloor \frac{|S| \cdot \Delta t}{\tau_{ref}} \right\rfloor \in O(|S|)$$

Despite this constraint, the global Lipschitz constant satisfies:

$$L_{globalS} = O\left(L_{global}^{|S|}\right)$$

and therefore sample complexity remains:

$$M = O\left(D|S|^2 \log(L_{global})\right)$$

Proof. A neuron can fire at most once per τ_{ref} interval. Over total time $T = |S| \cdot \Delta t$, this bounds $k_{\max} \in O(|S|)$. However, in the MIS proof (Theorem 2.2), causal sets grow as $|C_i^{(\ell)}| \propto k_{\max}$ across the sequence. Thus:

$$L_{\text{global}S} \leq \prod_{\ell} \left(|C_i^{(\ell)}| \cdot k_{\max}\right)^{|S|} \cdot \max\left(\frac{\bar{W}}{\delta}, \frac{\tau_s}{\delta}\right) \quad \leq (c \cdot k_{\max})^{|S|} = (c \cdot O(|S|))^{|S|} \quad = O\left(c'^{|S|}\right) \text{ for constant } c'$$

The exponential structure is preserved. □

Lemma 3.1 demonstrates that the refractory period does not induce inactivity in a manner that gets rid of the exponential $|S|$. Therefore, quadratic scaling is preserved.

Lemma 3.2 (Population Normalization - Gain Control)**.** *Let $N_\ell(\mathbf{V}) = \mathbf{V}/\sqrt{\varepsilon_\ell + \frac{1}{n_\ell}\sum_j V_j^2}$ be a divisive normalization operator with bounded Jacobian $\|J_{N_\ell}\| \leq G_\ell \leq 2$. Normalization ([16], [17]) modifies per-layer bounds by a constant factor:*

$$L_\ell^{norm} \leq G_\ell \cdot L_\ell$$

but preserves the exponential structure for sequences:

$$L_{globalS}^{norm} = O\left((G \cdot L_{global})^{|S|}\right) = O\left(c^{|S|}\right)$$

Therefore, sample complexity remains quadratic: $M = O(D|S|^2 \log(L_{global}))$. This aligns with cortical divisive normalization, demonstrating that biological gain control reduces constants but does not linearize scaling.

At first glance, this behavior appears to be a network saturation problem — something biological systems would avoid through population-level normalization. However, as shown in Lemma 3.2, the sequential manner in which these signals are integrated still preserves the exponential growth structure and quadratic scaling, despite normalization.

Proof of Lemma 3.2. The divisive normalization operator N_ℓ has bounded Jacobian $\|J_{N_\ell}\| \leq G_\ell \leq 2$. For a normalized layer $f_\ell^{\text{norm}} = N_\ell \circ f_\ell$, the chain rule gives $L_\ell^{\text{norm}} \leq G_\ell \cdot L_\ell$. Following Theorem 2.2 telescoping argument for sequence composition:

$$L_{\text{global},S}^{\text{norm}} \leq \prod_{\ell=1}^{|S|} L_\ell^{\text{norm}} \quad \leq \prod_{\ell=1}^{|S|}(G_\ell \cdot L_\ell) = \left(\prod_{\ell=1}^{|S|} G_\ell\right) \cdot \left(\prod_{\ell=1}^{|S|} L_\ell\right) \quad \leq 2^{|S|} \cdot L_{\text{global}}^{|S|} = (2 \cdot L_{\text{global}})^{|S|} = O(c^{|S|})$$

By Theorem 2.3, $M = O(D \cdot |S|^2 \cdot \log(c^{|S|})) = O(D \cdot |S|^2)$ since $\log(c^{|S|}) = |S| \log c$ absorbs into the quadratic term. The exponential structure is preserved, demonstrating that biological gain control reduces constants but does not linearize scaling. □

Because the brain is remarkably efficient at modeling language and learning long-range dependencies, we hypothesize that a third mechanism is present in biological neural nets that enables efficient learning by bounding uncontrolled excitation.

3.2 Necessity of Inhibitory Structures

Lateral inhibition linearizes the quadratic scaling induced by spiking dynamics, enabling the brain to efficiently learn long-range dependencies.

Lemma 3.3 (Inhibitory Sparsification and Telescoping MIS Bound)**.** *Let $\mathcal{N}$ be a network with L layers processing sequences of length $|S|$. Consider two inhibitory mechanisms ([18]):*

1. ***Per-bin WTA cap:*** *A hard limit s_ℓ on the number of active neurons per layer per time bin.*
2. ***Threshold boost:*** *Dynamic threshold adjustment*

$$\theta_\ell(t) = \theta_{0,\ell} + \beta_\ell(K_\theta * I_\ell^{\text{inh}})(t),$$

where I_ℓ^{inh} represents inhibitory input (modeling PV/SST interneurons). Then the following hold:

(i) ***Margin bounds:*** *$|C_i^{(\ell)}| \leq s_{\ell-1} k_{\max}^{(\ell-1)}$ where $k_{\max}^{(\ell-1)}$ is the maximum number of spikes per neuron in layer $\ell - 1$.*

(ii) ***Effective margin preservation:*** *The effective margin $\delta_\ell^{\text{eff}} > 0$ remains bounded across layers.*

(iii) ***Telescoping global Lipschitz bound:*** *$L_{globalS} = O\big(L_{global}^{|S|}\big)$ where L_{global} is the single-spike Lipschitz constant.*

(iv) ***Sample complexity:*** *$M = O\big(D|S|^2 \log L_{global}\big)$.*

Critical distinction: *Soft inhibitory mechanisms (WTA caps with $s_\ell = \Omega(|S|)$ or threshold boosts) reduce only the* constants*. To achieve* linear *scaling, a hard constraint $k_{\max} \le K = O(1)$ independent of $|S|$ is required.*

Lemma 3.4 (Linearization via Hard Per-Sequence Cap)**.** *Impose a* ***hard constraint*** *$k_{\max} \le K = O(1)$ independent of $|S|$. Then $M = O\big(D|S| \log L_{global}\big)$.*

Proof. Consider a neuron n_i^ℓ processing a sequence of $|S|$ inputs. The first input S_1 propagates through the resting network with local Lipschitz constant:

$$L_{\text{CausalPath1}} = \prod_{\ell=1}^{L} 2 \max |C_i^{(\ell)}| \max\Big(\frac{\bar{W}}{\delta}, \frac{\tau_s}{\delta}\Big).$$

Subsequent inputs encounter neurons with residual membrane potential. The causal set for S_2 depends on that of S_1, causing interdependence:

$$\big|T_{\text{output2}}(S_1, S_2) - T_{\text{output2}}(S_1', S_2')\big|_{L_\infty} \le (L_{\text{CausalPath1}} \circ L_{\text{CausalPath2}}) \|(S_1, S_2) - (S_1', S_2')\|_{L_\infty}.$$

Without inhibition, causal sets accumulate contributions from all previous inputs. Telescoping through $S_1, \ldots, S_{|S|}$ yields:

$$L_{\text{GlobalS}} = O\Big(\prod_{\ell=1}^{L} \max_i |C_i^{(\ell)}|^{|S|} \max\Big(\frac{\bar{W}}{\delta}, \frac{\tau_s}{\delta}\Big)\Big) = O(L_{\text{global}}^{|S|}).$$

Hard cap breaks the telescoping. Impose $k_{\max} \le K = O(1)$: each neuron fires at most K times total across the entire sequence. For any subset of inputs $\{S_1, \ldots, S_j\}$ with $j \le |S|$:

$$|C_i^{(\ell)}(S_1, \ldots, S_j)| \le s_{\ell-1} K = O(1).$$

The total spike count is capped, so the per-layer Lipschitz factor remains $L_{\text{layer},\ell} = O(1)$, and:

$$L_{\text{GlobalS}} = \prod_{\ell=1}^{L} L_{\text{layer},\ell} = O(L_{\text{global}}),$$

independent of $|S|$.

Sample complexity. The input diameter is $\text{diam}(\mathcal{X}) = |S| \cdot \text{diam}(\mathcal{X}_1)$, but the effective dimension is $D_{\text{eff}} = O(D_1)$ due to the spike constraint:

$$\log \mathcal{N}(\gamma, \mathcal{F}) \le D_1 \log\left(\frac{2L_{\text{global}}|S|\text{diam}(\mathcal{X}_1)}{\gamma}\right) = O(D_1 \log |S|).$$

Taking $\gamma \propto 1/\sqrt{M}$ yields $M = O(D_1|S| \log L_{\text{global}})$, linear in $|S|$. □

To fit this biological explanation into our analysis of spiking dynamics, we must understand whether our derivation extends from nLIF networks to LIF networks. We are actively researching this extension, but hypothesize that the same behavior holds: we can always choose a Δt for our rate encoding that is arbitrarily smaller than the leak time constant, effectively recovering nLIF-like behavior.

Discussion

The quadratic scaling of sample complexity with sequence length in nLIF networks highlights their inherent difficulty in modeling long-term dependencies. Architectural factors — most notably the strictly sequential integration of inputs drives this growth; normalization can shrink constants but does not alter the sequence-length exponent. Importantly, our analysis shows that bounding the size of each neuron's causal set is a direct route to improved sample efficiency as sequences lengthen. Future work should therefore explore implementations of inhibitory layers in SNNs. Lateral inhibition has shown promise in computer vision tasks [19], yet it remains to be tested in language modeling architectures.

Our analysis links common circuit phenomena to formal parameters that govern expressivity and efficiency in spiking networks. Inhibitory mechanisms, including effects consistent with PV- and SST-class interneuron activity, can reduce the size of active causal sets ($|C_i^{(\ell)}|$; see Lemma 4.3), yielding sparser effective population codes. Divisive normalization–like gain control can bound per-layer sensitivity via factors G_ℓ (Lemma 4.2), thereby reducing Lipschitz constants L_ℓ without altering the sequence-length exponent. Refractory and leak dynamics (Lemma 4.1) cap the per-neuron spike count $k_{\max}$, limiting dynamic range but leaving the quadratic dependence on $|S|$ unchanged unless an explicit hard cap is enforced.

These correspondences suggest that biologically plausible constraints help organize SNN dynamics within a Lipschitz-bounded regime that supports robustness and learnability. Together, these observations point toward the efficiency of biologically grounded, brain-inspired architectures, where inhibitory balance, normalization, and refractory dynamics jointly maintain robust and energy-efficient computation while preserving expressive capacity.

Scope note. All global Lipschitz and covering arguments presented here are proved for nLIF neurons. Extending to LIF will require establishing local Lipschitz continuity across reset surfaces, which we conjecture holds in the small-Δt regime with bounded leak.

Conclusion

We presented, to our knowledge, the first covering-number analysis of spiking architectures, establishing a quadratic sample-complexity bound for nLIF networks for monotone input sequences (MIS). This bound arises from the growth of causal sets across layers and time, which compounds global Lipschitz factors with sequence length. In this sense, sequential integrate-and-fire dynamics impose a structural limit on how spiking networks scale with long sequences.

Additionally, the analysis makes explicit how biologically plausible mechanisms map onto these formal parameters. Inhibitory interactions can sparsify active causal sets, divisive normalization can bound layer sensitivity and reduce Lipschitz constants, and refractory or leak dynamics can cap the per-neuron spike count. Together, these constraints shrink constants and, when a hard per-sequence inhibitory cap is enforced, can yield linear scaling in sequence length. This identifies concrete levers for architectural design and neuromorphic deployment.

While our results do not claim that spiking networks cannot capture long-range dependencies, they show that efficient sequence processing requires mechanisms that limit causal-set growth or otherwise regularize sensitivity. Assertions about frequency selectivity in language modeling remain hypotheses in this work and warrant targeted empirical tests.

All global Lipschitz and covering arguments here are proved for nLIF neurons. Extending these results to standard LIF models will require establishing local Lipschitz continuity across reset surfaces, which we conjecture holds in the small-Δt regime with bounded leak.

More broadly, the correspondences developed here suggest that biologically grounded constraints help organize spiking network dynamics within a Lipschitz-bounded regime that supports robustness and learnability. This framework points toward brain-inspired, energy-efficient architectures in which inhibition, normalization, and refractory dynamics jointly maintain expressive yet stable computation.

Author Contributions

W.F. conceived and performed the formal analysis, with S.H. contributing to conceptual design and biological contextualization. W.F. and S.H. jointly developed the methodological framework and discussed the theoretical implications. W.F. conducted the investigation, implemented analyses, and carried out mathematical derivations. S.H. provided supervision, critical feedback, and contextualized the biological relevance of the results. W.F. drafted the manuscript, and S.H. provided critical revision and editorial input. Both authors approved the final version of the manuscript.

References

[1] R.-J. Zhu, Q. Zhao, G. Li, and J. K. Eshraghian, "Spikegpt: Generative pre-trained language model with spiking neural networks," *arXiv preprint arXiv:2302.13939*, 2023.

[2] M. Schmitt, "Vc dimension bounds for networks of spiking neurons." in *ESANN*, 1999, pp. 429–434.

[3] P. Koiran and E. D. Sontag, "Vapnik-chervonenkis dimension of recurrent neural networks," *Discrete Applied Mathematics*, vol. 86, no. 1, pp. 63–79, 1998.

[4] Y. Chen, H. Liu, K. Shi, M. Zhang, and H. Qu, "Spiking neural network with working memory can integrate and rectify spatiotemporal features," *Frontiers in Neuroscience*, vol. 17, p. 1167134, 2023.

[5] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. M. McGinnity, "A review of learning in biologically plausible spiking neural networks," *Neural Networks*, vol. 122, pp. 253–272, 2020.

[6] B. L. Edelman, S. Goel, S. Kakade, and C. Zhang, "Inductive biases and variable creation in self-attention mechanisms," in *International Conference on Machine Learning*. PMLR, 2022, pp. 5793–5831.

[7] M. Schmitt, "On the sample complexity of learning for networks of spiking neurons with nonlinear synaptic interactions," *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 995–1001, 2004.

[8] W. Maass and M. Schmitt, "On the complexity of computing and learning with networks of spiking neurons," in *Electronic Proceedings of the Fifth International Symposium on Artificial Intelligence and Mathematics, http://rutcor. rutgers. edu/˜ amai/. Journal version to appear in Information and Computation*. Citeseer, 1998.

[9] A. M. Zador and B. A. Pearlmutter, "Vc dimension of an integrate-and-fire neuron model," in *Proceedings of the ninth annual conference on Computational learning theory*, 1996, pp. 10–18.

[10] D. Dold and P. C. Petersen, "Causal pieces: analysing and improving spiking neural networks piece by piece," 2025. [Online]. Available: https://arxiv.org/abs/2504.14015

[11] W. Maass and M. Schmitt, "On the complexity of learning for spiking neurons with temporal coding," *Information and Computation*, vol. 153, no. 1, pp. 26–46, 1999.

[12] M. Chen, X. Li, and T. Zhao, "On generalization bounds of a family of recurrent neural networks," *arXiv preprint arXiv:1910.12947*, 2019.

[13] W. Dorrell, M. Yuffa, and P. E. Latham, "Meta-learning the inductive bias of simple neural circuits," in *International Conference on Machine Learning*. PMLR, 2023, pp. 8389–8402.

[14] A. L. Hodgkin and A. F. Huxley, "The dual effect of membrane potential on sodium conductance in the giant axon of loligo," *The Journal of physiology*, vol. 116, no. 4, p. 497, 1952.

[15] ——, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[16] D. J. Heeger, "Normalization of cell responses in cat striate cortex," *Visual neuroscience*, vol. 9, no. 2, pp. 181–197, 1992.

[17] M. Carandini and D. J. Heeger, "Normalization as a canonical neural computation," *Nature reviews neuroscience*, vol. 13, no. 1, pp. 51–62, 2012.

[18] P. D. King, J. Zylberberg, and M. R. DeWeese, "Inhibitory interneurons decorrelate excitatory cells to drive sparse code formation in a spiking model of v1," *Journal of Neuroscience*, vol. 33, no. 13, pp. 5475–5485, 2013.

[19] X. Liu, "Inhibition snn: unveiling the efficacy of various lateral inhibition learning in image pattern recognition," *Discover Applied Sciences*, vol. 6, no. 11, p. 611, 2024.

[20] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

A Spiking Neural Networks

Definition 1. ***Spiking Neural Networks (SNN)*** *are biologically plausible neural networks that evolve continuously in time and emit discrete spikes whenever the membrane potential exceeds a preset threshold. Their dynamics follow:*

$$\frac{dV_i(t)}{dt} = \frac{1}{\tau_m}\big(V_{\text{rest}} - V_i(t)\big) + \frac{1}{\tau_s}\sum_j w_{ij} \sum_{t_j^k + \, t\Delta_{ij} \le t} H\big(t - t_j^k\big)\exp\Big(-\tfrac{t-t_j^k}{\tau_s}\Big)$$

Subject to the reset condition:

$$\textit{if } V_i(t^-) \ge \Theta_i, \quad V_i(t^*) = V_{\text{reset}}.$$

τ_m : Membrane time constant (leak rate).

$V_i(t)$: Membrane potential of neuron i at time t.

V_{rest} : Resting (leak) potential.

w_{ij} : Synaptic weight from neuron j to neuron i.

t_j^k : Time of the k-th spike from neuron j.

T_s : Synaptic delay before post-synaptic effect.

$H(\cdot)$: Heaviside step (spike) function.

τ_s : Synaptic time constant in $\alpha(s)$.

Θ_i : Firing threshold for neuron i.

$t\Delta_{ij}$: The time it takes for a current to reach a postsynpatic neuron from a pre-synaptic neuron.

V_{reset} : Reset potential after a spike.

$\sum_j w_{ij}$ External input current to neuron i from all neurons in previous layer.

A.1 Leaky Integrate & Fire Model

There are many biologically realistic neuronal models, but the Leaky Integrate & Fire Model (LIF) is the most commonly used due to its simplicity. The LIF simulates neuronal behavior by integrating incoming currents, applying a gradual leak over time, and emitting a spike when the membrane potential exceeds a set threshold. Unlike traditional artificial neural networks where neurons are updated synchronously, in the LIF model, neurons accumulate input over time and fire only when sufficient input is gathered to cross the threshold. The definition used for an SNN follows the same subthreshold dynamics of the membrane potential in an LIF model:

$$\frac{dV_i(t)}{dt} = \frac{1}{\tau_m}\big(V_{\text{rest}} - V_i(t)\big) + \frac{1}{\tau_s}\sum_j w_{ij} \sum_{t_j^k + t\Delta_{ij} \le t} H\big(t - t_j^k\big)\exp\Big(-\tfrac{t-t_j^k}{\tau_s}\Big)$$

At time t, if the pre-synaptic neuron j at time t_j^k reaches it's threshold, an instantaneous spike is triggered causing a jump in input current that then decays exponentially over the synaptic time constant τ_s. Writing this out:

$$I_i(t) = \frac{1}{\tau_s}\sum_j w_{ij} \sum_{t_j^k + \Delta_{ij} \le t} \exp\Big(-\frac{t - t_j^k - \Delta_{ij}}{\tau_s}\Big).$$

Each pre-synaptic spike at time t_j^k reaches neuron i only after delay $t\Delta_{ij}$[1], then produces a jump in current that decays exponentially with time constant τ_s. The delay is a programmable parameter that increases the expressivity of a spiking neuron [8]. However, for the primary analysis using the covering-number, we treat this as a fixed parameter.

A.1.1 Spike Trains

Spike trains encode each input by converting its value into the timing and amplitude of Dirac delta pulses, which are then injected into the network to initiate the spiking process. The spike train is derived through a procedure that maps real-valued inputs to a tensor of Dirac delta functions.

Consider:

$$\begin{aligned}
\text{Data Matrix \& Time Grid:} \quad & X \in R^{N\times T}, \quad t\Delta > 0, \quad t_k = k\,t\Delta, \\
& \text{where } t\Delta \text{ is the inter-spike delay and } t_k \text{ is the } k\text{th time point,} \\
\text{Threshold Encoding:} \quad & S_i[k] = \begin{cases} 1, & x_i[k] > \theta_i, \\ 0, & \text{otherwise,} \end{cases} \quad i = 1,\dots,N,\ k = 0,\dots,T-1, \\
\text{Discrete Spike Vector:} \quad & S[k] = \big[S_1[k],\,\dots,\,S_N[k]\big]^T \ \in\ \{0,1\}^N, \\
\text{Continuous-Time Spike Train:} \quad & s_i(t) = \sum_{k=0}^{T-1} S_i[k]\,\delta\big(t - k\,t\Delta\big), \quad i = 1,\dots,N, \\
\text{Spike-Train Vector:} \quad & \mathbf{S}(t) = \big[s_1(t),\,\dots,\,s_N(t)\big]^T, \\
\text{Input Current:} \quad & I(t)_1 = \mathbf{W_1}\,\mathbf{S}(t).
\end{aligned}$$

This process achieves two goals[2]. First, it binarizes the raw inputs into a sequence of 0/1 spikes. Second, it embeds those spikes in continuous time by replacing each "1" in the discrete tensor with a Dirac delta at the appropriate moment. Concretely, for neuron i we write

$$s_i(t) \;=\; \sum_{k=0}^{T-1} S_i[k]\,\delta\big(t - k\,t\Delta\big),$$

so that a "1" at index (i,k) becomes a δ-spike at time $t = k\,t\Delta$. From the network's point of view, this is equivalent to feeding it a stream of binary tensors $S[0], S[1], \dots, S[T-1]$. By casting those tensors as continuous spike trains, we naturally capture all temporal dependencies—so the LIF membrane and synaptic decay dynamics require no extra bookkeeping, because the δ pulses encode precisely when inputs arrive.

Once the membrane potential reaches the threshold θ, the neuron emits a spike and its potential is reset:

$$\text{if } v(t) \geq \theta \quad \Longrightarrow \quad \begin{cases} \text{spike is emitted,} \\ v(t) \leftarrow v_{\text{reset}}. \end{cases}$$

Importantly, the entire membrane potential process is initiated by the input state *spike trains*: at each time step k, the threshold-ed vector $S[k]$ determines which neurons fire—every neuron with $S_i[k] = 1$ emits a Dirac-δ spike, kick-starting the propagation of signals through the network.

[1] A time delay represents the time for a signal to be transmitted from a pre-synaptic neuron to a post-synaptic neuron. These are constant throughout the network and mimic time to travel in actual neuronal communication

[2] Threshold encoding is for illustrative purposes and is not Lipschitz continuous. Do not use this technique in the function composition in Section 3.

A.1.2 Model Output

In order to actually use these networks it is important to understand how to retrieve output values from them. The output layer is monitored differently depending on the task being performed. For Example, during classification tasks, we monitor the output layer over a fixed time window T. Let

$$n_i \;=\; \sum_{t=1}^{T} s_i(t),$$

denote the spike count of output neuron i in the window T, where T represents the total elapsed time in the network. We map these rates to class probabilities with a softmax:

$$p_i \;=\; \frac{\exp\big(n_i\big)}{\sum_{j=1}^{C} \exp\big(n_j\big)}, \qquad i = 1, \ldots, C.$$

For sequence–generation tasks (e.g. language modeling) we take a snapshot of the output layer's membrane potentials at the end of the time window. This moment is when the last token propagates entirely through the network.
Let

$$Y' \in R^d$$

be this membrane potential vector and

$$W \in R^{|V| \times d}$$

the embedding matrix whose v-th row represents vocabulary token v. The logit for token v is

$$\text{logit}_v \;=\; Y' \, W_v^{\top}.$$

The probability assigned to token i is obtained with a soft-max:

$$P(i) \;=\; \frac{\exp\big(\text{logit}_i\big)}{\sum_{v \in V} \exp\big(\text{logit}_v\big)}.$$

Once the spiking outputs are mapped to a probability distribution, training proceeds almost exactly as in a ANN.

A.1.3 Surrogate Gradient Descent

The commonly used Heaviside activation function is not differentiable, making it impossible to perform gradient descent. A commonly used method is to use a differentiable function which approximates the SNN activation function and use backpropagation through time (BPTT) to learn the weights of the network.

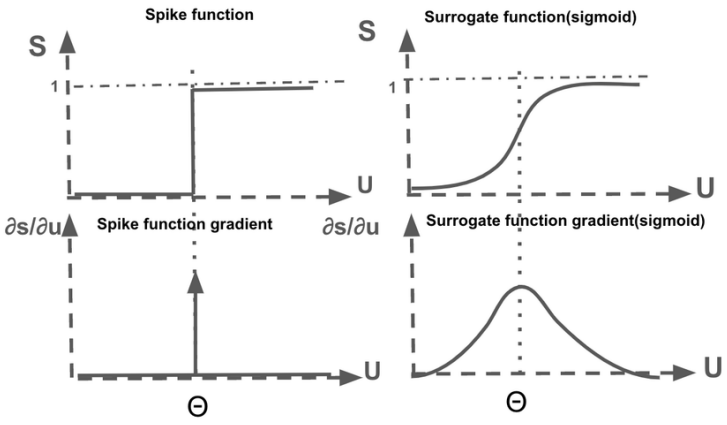

Figure 1: Visualization of Different Commonly Used Surrogate Gradients [20]

This allows the cross-entropy loss gradient to propagate through the network and update the weight values. Some networks also allow for learnable decay functions, but in the traditional LIF the decay rates and thresholds are fixed.

B VC Dimension Proof Sketches for Supplementary Results of SNNs

We give informal sketches of the VC-dimension bounds of SNNs that help motivate our work. These sketches are not intended as full formal proofs (see the original references for complete arguments), but rather as a reader's guide to the key ideas and constructions.

Theorem B.1. *For each n one can construct a feedforward spiking-neuron network (SNN) N with $O(n)$ edges whose VC dimension is $\Omega(n^2)$. This remains true even if all synaptic weights and firing thresholds are held fixed.*

Proof Sketch. Let

$$R = \{1, 2, \ldots, n\}, \quad D = R \times R = \{(k, i) \mid k, i \in R\},$$

and fix any binary labeling stored in the $n \times n$ matrix $B = [b_{k,i}]$. Encode each row k by the delay

$$d_k \;=\; \sum_{j=1}^{n} b_{k,j}\, 2^{\,j-n-1}.$$

Inputs. The network has $2n + 1$ input neurons: $\{X_1, \ldots, X_n\}$ (row selectors), $\{Y_1, \ldots, Y_n\}$ (column selectors), and Z (global reference). On presentation of (e_k, e_i), exactly X_k, Y_i, and Z fire at $t = 0$. The X_k spike then arrives at the first module M_n after delay d_k.

Module M_m (bit-extraction). Each module M_m $(m = n, n-1, \ldots, 1)$ does the following:

- *IN1:* data spike at
$$t = r_m \;=\; \sum_{j=1}^{m} b_{k,j}\, 2^{\,j-n-1}.$$
- *IN2:* reference spike from Z at $t = 0$.
- *OUT2:* if $b_{k,m} = 1$, emits at $t = r_m$.
- *OUT1:* emits at
$$t = (2 + 2\beta + 2\gamma) + r_{m-1}, \quad r_{m-1} = \sum_{j=1}^{m-1} b_{k,j}\, 2^{\,j-n-1},$$
and forwards this to M_{m-1}.

Coincidence detection. For each column i, neuron C_i receives:
$$\mathrm{OUT2}_i \quad \text{at } t = r_i, \qquad Y_i\text{-spike delayed to } t = r_i.$$
With unit weights and threshold 2, C_i fires exactly at $t = r_i$ if and only if $b_{k,i} = 1$. No other C_j can fire.

Thus each (k, i) is correctly labeled by whether C_i spikes, and since we can choose all n^2 bits arbitrarily via the delays $\{d_k\}$, the network shatters D using only $O(n)$ edges [11]. □

While these constructions are stated for non-leaky neurons, they rely solely on programmable delays and hence carry over to leaky membranes without loss of generality. Moreover, since adding programmable weights and thresholds cannot reduce expressivity, the same $\Omega(n^2)$ lower bound holds for fully programmable SNNs. The authors also show that any SNN with L edges, programmable weights, delays, and thresholds, and depth D satisfies
$$\mathrm{VCdim}(N) \;=\; O\big(L\, D \log(LD)\big).$$
In the case of a network whose depth scales with its width ($D = \Theta(L)$), this upper bound simplifies as follows:
$$O\big(L\, D \log(LD)\big) \;=\; O\big(L \cdot \Theta(L)\, \log\big(L \cdot \Theta(L)\big)\big) \;=\; O\big(L^2 \log(L^2)\big) \;=\; O(L^2).$$

C Causal Pieces & Local Lipschitz Continuity

This section provides an overview of the work done by Dold et al. Our work is focused on extending the implications of their work thus key points which were not defined in the main paper and the key local Lipschitz continuity proof sketch are in this section.

Definition 2. ***Non Leaky Integrate-&-Fire (nLIF)*** *is a version of the LIF model in which there is no membrane leak. Specifically:*
$$\frac{dV_i(t)}{dt} = \frac{1}{\tau_s} \sum_j w_{ij} \sum_{t_j^k + t\Delta_{ij} \le t} H\big(t - t_j^k\big) \exp\Big(-\tfrac{t - t_j^k}{\tau_s}\Big)$$

Theorem C.1 (Local Lipschitz Continuity of nLIF Spike Times)**.** *Let $N_0 \in N$ and consider a single non-leaky integrate-and-fire (nLIF) neuron with input spike times*
$$a, b \;\in\; P\big[C_1^{(1)}\big] \;\subset\; R^{N_0},$$
where $C_1^{(1)} \subset \{1, \ldots, N_0\}$ is its causal set (i.e. the indices of all pre-synaptic spikes that occur before the output spike), synaptic time constant $\tau_s > 0$, threshold ϑ, and weights satisfying
$$\sum_{j \in C_1^{(1)}} W_{1j} - \vartheta \;=\; \delta \;>\; 0, \qquad \|W_{1j}\| \le \bar{W} \quad (\forall j \in C_1^{(1)}).$$

Denote by

$$t_1^{(1)}(x) \;=\; \tau_s \ln\Big(\frac{\sum_{j\in C_1^{(1)}} W_{1j}\, e^{\,x_j/\tau_s}}{\sum_{j\in C_1^{(1)}} W_{1j} - \vartheta}\Big)$$

the output spike time given inputs $x \in R^{N_0}$. Then on the causal piece $P[C_1^{(1)}]$,

$$\big\|t_1^{(1)}(a) \;-\; t_1^{(1)}(b)\big\|_{L\infty} \;\le\; 2\,|C_1^{(1)}|\;\max\Big(\tfrac{\bar W}{\delta},\,\tfrac{\tau_s}{\delta}\Big)\;\|a-b\|_{L\infty}.$$

Moreover, as long as one remains in a region where $\sum_{j\in C'} W_{1j} - \vartheta > 0$, continuity is preserved; if the causal set becomes empty the spike time can jump discontinuously.

Proof Sketch.

Within a causal piece, differentiability. On $P[C_1^{(1)}]$, the causal set $C_1^{(1)}$ is fixed, so $t_1^{(1)}(\cdot)$ is a composition of logarithm, exponential, sums and quotients of affine functions of the inputs and weights. Hence it is smooth there.

One checks directly from

$$t = \tau_s \ln\big(T\big), \qquad T = \frac{\sum_{j\in C} W_j e^{\,x_j/\tau_s}}{\sum_{j\in C} W_j - \vartheta},$$

that for each input coordinate x_k,

$$\big|\partial t/\partial x_k\big| = e^{-t/\tau_s}\,\frac{W_k\, e^{\,x_k/\tau_s}}{\sum_{j\in C} W_j - \vartheta} \;\le\; \frac{\bar W}{\delta},$$

and for each weight W_k,

$$\big|\partial t/\partial W_k\big| \;\le\; \frac{\tau_s}{\delta}.$$

By the multivariate mean-value theorem in the ℓ_∞–norm,

$$\|t(a)-t(b)\|_{L\infty} \;\le\; \sup_{\xi\in \mathrm{conv}\{a,b\}} \|\nabla t(\xi)\|_1\,\|a-b\|_{L\infty} \;\le\; 2\,|C|\;\max\Big(\tfrac{\bar W}{\delta},\tfrac{\tau_s}{\delta}\Big)\,\|a-b\|_{L\infty}.$$

Here the factor $2\,|C|$ arises since there are $|C|$ partials in the inputs and $|C|$ in the weights.

When the input or weights cross into a different causal set C', continuity persists as long as $\sum_{j\in C'} W_j - \vartheta > 0$. If one enters a region where no pre-synaptic input suffices to reach threshold, the output spike time jumps to ∞.

This completes the proof sketch of the work done by Dold et al. For a more formal overview of their work see the original paper [10] □

D Lipschitz Function Composition Example

Example D.1. Consider a vocabulary M where each $m_i \in M$ is a word in an embedding space s.t. m_i is a R^D tensor. WLOG, these words are adjectives with positive or negative sentiment, such as (Happy, Sad, Good, Bad,...). The goal is to build a network that classifies these words based on their associated sentiment.

Using Time to First Spike (TTFS), a Lipschitz continuous spike encoding function can be created to encode the spike trains[1]. For simplicity a simple normalization TTFS function is chosen where

$$t(m_i) = \frac{m_i^j - \min(M^j)}{\max(M^j) - \min(Mj)} \;* T,\; T = \text{Time Constant}$$

[1]Not all spike encoding functions are Lipschitz continuous without added assumptions. Many like threshold encoding are not without added assumptions about the input space.

This encodes an input tensor where all input layer neurons will fire at some time $t(m_i)^j$. We assume that the $t(m_i)^j$ cells are close enough together s.t. that one input does not trigger a neuron to spike multiple times, so as not to invalidate our SIS setup. Clearly, this rate encoding scheme is Lipschitz continuous. Since the nLIF spike train inputs are bounded by $\frac{\delta}{L_{Global}}$ to ensure Lipschitz continuity with the M input space we construct a bound for our input samples s.t. $|t(m_i)-t(mi^{'})|_{L\infty} \leq L_m|m_i-m_i^{'}|_{L\infty} \leq \frac{\delta}{L_{global}} \Rightarrow |m_i-m_i^{'}|_{L\infty} \leq \frac{\delta}{L_{global}*L_m}$ where L_m is the spike train encoding Lipschitz value and L_{global} is the global Lipschitz value for the nLIF w.r.t spike times and weights of the nLIF. The goal of this Example is to classify positive and negative sentiment from single input spikes. Let the output layer consist of two neurons, "pos" and "neg," with spike times $T_{\text{pos}}, T_{\text{neg}}$. Define

$$r_i = \frac{1}{T_i}, \quad i \in \{\text{pos}, \text{neg}\},$$

$$p_{\text{pos}} = \frac{e^{\beta\, r_{\text{pos}}}}{e^{\beta\, r_{\text{pos}}} + e^{\beta\, r_{\text{neg}}}}, \quad p_{\text{neg}} = 1 - p_{\text{pos}},$$

and classify by

$$\hat{y} = \arg \max_{c \in \{\text{pos},\text{neg}\}} p_c.$$

This decoding function converts these spike times back to outputs. This is an application of the softmax in the output and this decoding function is Lipschitz with some bound L_o. Since the nLIF SNN is able to be decomposed into a set of Lipschitz continuous functions, the entire network is Lipschitz continuous s.t. for some $m_i, m_j \in M$ where

$$|m_i - m_j|_{L\infty} \leq \frac{\delta}{L_m * L_o * L_{global}}$$

Then

$$|F(m_i) - F(m_j)|_{L\infty} \leq L_m * L_o * L_{global}|m_i - m_j|_{L\infty}$$